<u>Next AUG Meeting</u>

Sunday, October 16th, 1988 at 2pm

(Doors open at 1pm, meeting starts at 2pm sharp)

AUG meetings are held in the Rotunda at Monash University
Wellington Road, Clayton Melways map 70 reference F10 and map 84A

Amiga Users Group Inc, PO Box 48, Boronia, 3155, Victoria, Australia

AMIGA™ Users Group

Who Are We?

The Amiga Users Group is a not-for-profit association of people interested in the Amiga computer and related topics. With over 1000 members, we are the largest independent association of Amiga users in Australia.

Club Meetings

Club meetings are held at 2pm on the third Sunday of each month in the Rotunda at Monash University, Wellington Road, Clayton. Details on how to get there are on the back cover of this newsletter. The dates of upcoming meetings are:

Sunday, October 16th at 2pm
Sunday, November 20th at 2pm
Sunday, December 18th at 2pm

Production Credits

This month's newsletter was edited by Peter Jetson. Equipment and software used was: Non-descript Taiwanese PC Clone computer, Brother HR-40 printer, Brother HL-8 printer, Gemini 10x printer, Wordstar, Fancy Font and Grabbit.

Copyright and Reprint Privileges

Amiga Workbench is Copyright © 1988 by the Amiga Users Group Inc. Articles herein that are copyrighted by individual authors or otherwise explicitly marked as having restricted reproduction rights may not be reprinted or copied without written permission from the **Amiga Users Group** or the authors. All other articles may be reprinted for any **non-commercial** purpose if accompanied by a credit line including the original author's name and the words "Reprinted from **Amiga Workbench**, newsletter of the **Amiga Users Group**, PO box 48, Boronia, 3155".

Contributions

Articles, papers, letters, drawings and cartoons are actively sought for publication in Amiga Workbench. Please submit your contributions on disk, since that means they don't have to be re-typed! All disks will be returned! Please save your article in **text-only** format (If it can be loaded by ED, it is text-only). Absolute deadline for articles is 16 days before the meeting date. Contributions can be sent to: The Editor, AUG, PO Box 48, Boronia, 3155.

Membership and Subscriptions

Membership of the Amiga Users Group is available for an annual fee of $20. To become a member of AUG, fill in the membership form in this issue (or a photocopy of it), and send it with a cheque for $20 to:

Amiga Users Group, PO Box 48, Boronia, 3155

Public Domain Software

Disks from our public domain library are available on quality 3.5" disks for $8 each including postage on AUG supplied disks, or $2 each on your own disks. The group currently holds over 200 volumes, mostly sourced from the USA, with more on the way each month. Details of latest releases are printed in this newsletter, and a catalog disk is available.

Member's Discounts

The **Amiga Users Group** negotiates discounts for its members on hardware, software and books.

Currently, **Technical Books** in Swanston Street in the city offers AUG members a 10% discount on computer related books, as does McGills in Elizabeth Street. Just show your membership card. Although we have no formal arrangements with other companies yet, most seem willing to offer a discount to AUG members. It always pays to ask!

Back Issues of Newsletter

All back issues of Amiga Workbench are now available, for $2 each including postage. Note that there may be delays while issues are reprinted. Back Issues are also available at meetings.

AmigaLink - Our Bulletin Board System

The Amiga Users Group operates a bulletin board system devoted to the Amiga, using the Opus message and conferencing software. AmigaLink is available 24 hours a day on (03) 792 3918, and can be accessed at V21 (300bps), V22 (1200bps), V23 (1200/75bps) or V22bis (2400bps) using 8 data bits, 1 stop bit and no parity.

AmigaLink is part of a world-wide network of bulletin boards, and we participate in national and international Amiga conferences. AmigaLink has selected Public Domain software available for downloading, and encourages the uploading of useful public domain programs from its users. AmigaLink is FidoNet node number 3:631/324.

Newsletter Advertising

The Amiga Users Group accepts commercial advertising in Amiga Workbench subject to the availability of space at these rates:

Quarter page	$20
Half page	$40
Full page	$70
Double page spread	$120

These rates are for full-size camera-ready copy **only**. We have no photographic or typesetting facilities. Absolute deadline for copy is 16 days before the meeting date. Send the copy and your cheque to: The Editor, AUG, PO Box 48, Boronia, 3155, Victoria.

Amiga Users Group Committee

Co-ordinator:	Bob Scarfe	??? ????	Kensington
Vice Co-ord:	Lester McClure	233 5664	Mt Waverley
Meeting Chair:	Ron Wail	848 7283	Doncaster
Secretary:	John Elston	375 4142	Moonee Ponds
Treasurer:	Donna Heenan	729 2327	Bayswater
Membership:	Neil Murray	794 5683	Dandenong
Purchasing:	Bohdan Ferens	792 1138	Dandenong
Book Library:	Joan Wood	580 7463	Aspendale
Disk Library:	Craig Hutchison	578 3962	Carnegie
Editor:	Peter Jetson	762 1386	Boronia
Committee:	Doug Myers	428 0682	Abbotsford

Wot I lernt at da baginers C SIG
by Mal Woods

The more recent versions of C for the Amiga are approaching the ANSI standards for C.

Working through one of the examples in the public domain tutorial, I found that Lattice C Version 4.0 gave me a warning error at the beginning of a function. I had seen a similar error before when a main() wasn't declared as void, so I assumed that is what I had to do with this function. I inserted the keyword 'void' in front of the function and set it compiling. Now I had a fatal error. I presented this problem to the people who attended the meeting and with the help of Peter Jetson and a quick visit to Eric Salter in the Advanced C SIG we had the answer.

In ANSI C, functions should be defined before they are called. If they are not, then the compiler assumes that the result of the function will be an integer. If the function when found is undefined, a warning error will be generated. If it returns a different type, then a fatal error is generated.

Arrays in C start at 0 and go up to one less than the number in the definition.

Writing a small program for one of the exercises in the tutorial, I had to declare arrays with 10 elements to store simple data in. Fine, I said.

I remember reading that arrays in C start at 0, so if I want 10 elements then I assumed I needed to define the array like so - 'array[9]'. So I started my program running and the window seemed to lock up. A fair bit of what little memory I had had disappeared and the rest of the system ran very very slowly. I tried doing a break command from another window and nothing happened. I checked the manual, break should work, but it didn't.

I had to reboot and try it again. I checked the program and found nothing wrong with it. I was very confused. I commented out the first loop and the program ran successfully with garbage in the first array which was the loop I removed. Both loops looked the same; only defining the array with a different value.

When in doubt, put in a printf statement, they always say. So I put one in the first loop, printing the array element currently being modified and uncommented it. To my surprise I got 01234567891 23456789 continuous. I set the value I was placing in the first array to 2, ran the program and got 012345678923456789 continuous. Strange, the loop started again from the value I was placing in the array. Since I had a printf statement in the program, I could do a CNTRL C and Lattice asked me nicely if I would like to abort the program. I said I would.

By this time it was about 2am, so I decided to leave the problem to the gurus at the SIG. It was solved by a junior guru in the front row who found the error as soon as it appeared in the screen. I won't make that mistake again.

You too can have your (simple) C problems solved at the Beginner's C SIG. Hope to see you there.

===

Faery Tale (A Lateral Review)
by PLAMB

This review is in response to an earlier (and in my opinion somewhat unfair) review of Faery Tale by Bob Scarfe in the October 1987 issue of Workbench. In his review, Bob praises the 3D graphics and the large and varied scenery of this game and justifiably so. However he later says that "there doesn't seem to be any logic in this interesting waste of time...". I believe his impression was formed by trying to play this game in a brute force manner (ie. with a hack and slash mentality). Faery Tale is indeed a "very sophisticated little visual story" and it requires a lateral approach to solving many of the problems the adventurer is faced with (hence the title of this review).

In the following paragraphs, I'll outline two simple strategies to stay alive long enough in Faery Tale to enjoy the 3D graphics and see the surrounding countryside (which is really very interesting). It is possible to search the buildings in the home town for some minor equipment (use the look option to see hidden items), however these are generally of a trivial nature and not much help in the long term.

The first requirement is to acquire a better weapon than the dirk you start with. This weapon does

minimal damage to the badies - because they come (usually) in fours, they invariably end up killing you if you only have the dirk. Note that the right mouse button activates the fighting arm -in conjunction with the compass setting, this determines which direction you face when you fight. The left mouse button held down in conjunction with the compass direction selected moves you in that direction.

The best method of staying alive and getting a weapon (a mace is better than a dirk, and a sword is better than a mace, a bow can be useful if you can keep the badies at a distance, but you generally find that you run out of arrows at a critical time) is to follow the road east out of the town and follow it southwards when it branches. Continue to follow it southwards/SE (there is one branch to the west, ignore that for now). After a little while you will come to an iron fence that runs along the road for a way. This fence surrounds a graveyard. Follow the fence around until you find gate into the graveyard. Go inside and go to the opposite side of the graveyard away from the gate.

If you haven't dawdled on your way to the graveyard, the badies will arrive just after you're inside the fence and the important thing is that they are on the other side of the fence, and they can only attack you one at the time across the fence (and they don't seem to be able to do that very well). Just keep fighting them until one dies and then "take" his weapon when the others are not attacking you. You can move away and then back again in order to lure the other badies away from the body. Normally the badies also carry booty, ie. gold, vials of healing potion, keys to unlock locked doors, other magic items, etc. Note that a bird totem gives you a birds eye view of your surrounding area (like a map). This can be helpful in deciding which way to go next.

I usually hang around the graveyard for a while. Take a walk around the inside perimeter - by the time you get back, the old badies are gone and you get a new bunch to fight. They are sitting ducks while they are on the other side of the fence from you. (Note that some of the badies can pass through material objects! If you come across these types, well, I'll leave you to find your own solutions.) I stay at the graveyard until I build up my bravery points (by killing the badies), this makes it easier for me to kill the badies in future. When my bravery score is above 60 and I have a sword (not all badies carry swords), I quit the game saving it to disk. This allows me to start my next adventure at the point I saved the game (ie. saves me time in not having to build up a strong character each time).

The second method for staying alive requires a lot of skill and some luck. This procedure is for the hack and slashers amongst you, a real brute force method, no finesse involved at all and suitable for those who believe they can think better with their "mouse hand" than with their "grey matter".

Firstly, search the village. You should find at least one healing vial - if you're lucky you might find more. You will need these to heal yourself during your encounter with the badies. Wander out of the village to the east, but don't go too far. When the badies arrive (if you only get one, it does

happen but very rarely, you should go out and buy a tatts ticket) suspend the game and examine them for weapons. Be wary of those armed with swords and maces, never stand still if any are armed with bows and arrows. Examine the countryside for possible positions (trees, boulders etc) which you can use to prevent the badies from surrounding you. It is possible to lead the badies back to the village and fight them in the lanes between the buildings - they can only attack you one at the time there.

When you restart the game, position yourself to attack your chosen target and keep on the move. Alternatively, fight and move or the badies will surround you and you'll be dead meat. Keep an eye on the state of your health and use the healing vials as necessary. If you are lucky enough to have only skeletons attack you, you should be able to overcome them with only a moderate amount of skill (they are easier to kill than the human bandits and other nasties). Remember to search their bodies for goodies and arm yourself with the best weapon available to you.

Thereafter you just wander around the countryside trying to find the talisman. There are interesting places to see and interesting things to do on the way. Try out the magic items you find in different situations - they do some snazzy things. Best of luck in your adventuring and remember if you come across a new situation, stop and try some lateral thinking before rushing in.

I find this a very interesting and entertaining game, but then again I consider myself a bit of an adventurer. Do you?

===

AMIGA Assembly Language Programming - A Book Review
by Lester McClure

The first thing that puzzled me about this book was 'who is it written for?'. The author quite clearly sets out to teach the concepts of assembly language programming, starting with a simple introduction to binary number systems and continuing through to a complex program in assembler for the Amiga computer. This is quite a lot to cover in a single publication, but I think the author has succeeded.

What initially had me intrigued though, was who would be learning to program in assembler on the Amiga. I had assumed that most people interested in assembly language for the Amiga would already have a pretty good grounding in assembler and binary code concepts from work with other, much simpler microprocessor systems. While I was part the way through reading this book, something happened which changed my mind. I attended a short course on 'Introduction to the IBM PC' on which I studied the architecture of the 8088 microprocessor and the BIOS programming environment. I was not impressed! There was a general structure of awkwardness, a peculiar set of register interrelationships and the concept of Address Segmentation I found very limiting and confusing. I have programmed in assembler on the older 8 bit Intel microprocessors and had expected the 8088 to be a logical and tidy extension. I was wrong. The final outcome was that I now consider the Amiga with its

68000 microprocessor a good environment in which to learn to program in assembler language. I guess that the author had come to this conclusion before putting the effort into this book, so I will devote the rest of this article to a breakdown of the approach he has taken in introducing 68000 assembly language programming on the Amiga. Incidentally, the copy I reviewed came from our very own AUG book library.

The author (Jake Commander) introduces his book by stating the rewards that can be gained from programming any machine in assembler language - you can produce programs which have the most efficient implementation for that machine, and to be able to do it demonstrates a mastery over the computer and its operating system. The Amiga, with its wealth of useful routines already in ROM and the well documented requirements and parameters for each routine make it very easy to write powerful programs in assembler, without 're-inventing the wheel'.

Chapters 1 and 2 cover most of the basic terms and concepts of programming a computer at the most fundamental level. These include machine code vs. assembler, programs in the form of a stored sequence of instructions or opcodes, program counter, registers, assembler mnemonics, editor for preparation of source code, object code, addressing modes, assembler directives etc. If these seem pretty basic to you, I agree but it gives a good idea just how thorough the author has been in introducing assembly language programming to the complete novice.

Chapter 3 is a profile on the Motorola MC68000 microprocessor with quite a detailed description of the internal register set and its functions. It also includes a brief description of the chip from a hardware point of view, with a summary of the physical pin connections. This leads on to a thorough discussion on addressing modes in Chapter 4 and the the 68000 instruction set in Chapter 5. All instructions are listed in alphabetical order with a description of the syntax and the flags affected for each operation. This is the chapter where you would most likely leave a permanent bookmark.

At this stage we are about one third of the way through the book and nothing specific has been said about the Amiga. This changes in Chapter 6, which gives one of the best breakdowns of the structure of an Amiga system I have ever read. It summarises the Amiga programming environment as a set of interconnecting hardware and software parts - not just pieces of hardware like a keyboard, screen and printer but also pieces of pre-written software (such as the executive disk operating system and Intuition routines) which interconnect with each other and the hardware. Although when the book was written the A1000 was the only Amiga available, most of the text, except for loading Kickstart, applies to other models. There is an excellent description of processes and tasks, Exec, AmigaDOS and how the device and library modules all fit together in a set hierarchy.

In Chapter 7 the software interface to the Amiga libraries is outlined with particular reference to the concept of system structures which are used by many routines. The usefulness of standard 'include' files to define structures and equates is stressed and a brief rundown is given for some of the more

important ones. This section also covers opening libraries, calling routines within a library and gives a summary of the standard available libraries.

The next chapter describes the standard Amiga macro-assembler development system as available from Commodore-Amiga. This chapter is somewhat dated and there are now many alternatives, such as the assembler/linker from the Aztec C programming package and even public domain offerings (Fish disk #81 and #110). I would not consider using Ed or Edit (see my article in 'Workbench' August 1988) editors as there are better alternatives.

Chapter 9 discusses the traditional program design process - Edit, Assemble, Test then loop back to Edit until a complete working program is developed. It is, however, pleasing to see such a chapter included because quite often not enough thought is put into the specification and program design phase BEFORE coding begins. Chapter 10 takes a cursory look at alternatives to assembly language before moving on to some example programs in Chapter 11. These form a useful start and are probably worth typing in as a test of your development environment. For more advanced examples and programming techniques I suggest you investigate the public domain disks where there is quite a lot written in assembler despite the overall popularity of C.

The last two chapters introduce 'number-crunching', firstly with a closer look at the binary numbering system and then with a calculator program example.

The example runs for 40 pages in the book and I wasn't prepared to type it in to see if it would work, although I believe from another reference to this book that it does. A list of library routines forms an appendix to the book, with the parameters required for each entry and the library that contains it. While this information is useful, it is out of date as it covers V1.1 release only.

There are alternatives to this book, such as 'COMPUTE's Amiga Machine Language Programming Guide' and the Abacus publication 'Amiga Machine Language' which are more up-to-date. Currently they are more readily available than Amiga Assembly Language Programming (out of stock everywhere), but I believe its strong point is that although it is directed at the complete novice, it is useful for the experienced programmer. If a second edition is ever released I'll go out and buy it.

===

Ed
by John Pocock

Pat Huevel, our Beginners SIG maestro, made a remark at the last meeting that it was some months since he last used the command 'ed'. I wondered how he could possibly manage without it. For me, it is the 'open sesame' for just about everything that happens on my Amiga.

Because typing can be such a drag at times, I have devised many short cuts to reduce that effort to the absolute minimum. Number one is the reduction of all the most used 'C' commands to one letter, for example:-

        d for diskcopy
        q(quack) for diskdoctor
        k for copy
        f for format
            and so on.

All letters of the alphabet are used, except one, and some are used twice. 'X' of course brings us to the execute files and this is where 'Ed' really shines. For instance, let's have a look at our startup-sequence. I type 'x oh' to give me the startup-sequence for df0:, I type 'x od' for df1:s/startup-sequence and 'x ow' for for dh0:s/startup-hd.

Now 'od' is the name of the execute file, so to create it I type 'ed od'. Amiga immediately gives me a brand new file, and in it I type:

        'ed df1:s/startup-sequence'

then 'esc', then 'x' then press 'return', and the new file is ready to go. How would you feel about typing 'copy df0:devs/system-configuration df1:devs/system-configuration to transfer 'preferences'. A very neat operation is achieved by merely typing 'x hd' so, why do it the hard way? Now, how about a diskcopy program? Type 'x dd'.

To make the execute file type 'ed dd', and in the new file type:-

        k df0:c/d ram:
        k df0:c/x ram:
        cd ram:
        d df0: to df1:
        x df0:dd

then, to save it, type 'esc x' and then press 'return'.

This program resets itself after each disk is copied, and the same can be done with a format file. You need only press 'return' after reading the message and inserting the disks. What could be simpler?

To make these files always accessible it's advisable to place them in the 's' directory. This will ensure that Amiga will find it.

That's all for now; if you'd like more, a little persuasive feedback will do the trick.

===

Midi and the Amiga in Australia
by Edward Borland

So far not many people have had a lot of reason to whinge about our machine, yet for musicians, the situation is completely different. I now am starting to wonder why I just didn't go and buy an Atari to start with.

I have been very pleased with my A1000 in terms of word processing, games and graphics, and its ability to run several programs at once (great with a modem!). And yet it seems a child could keep better time than the Amiga when it comes to sequencing synthesizers. I have tried Pro Midi Soundscape, Deluxe Music, Dynamic Drums and Sonix (Musicraft). I'm using a Yamaha RX-15 drum machine and a DX-11 multitimbral synth. Yet when I try to synchronize my drum machine to the Amiga through the midi clock, playing the song only results in the keyboard's tempo drifting away from the drum machine. I'm using the midi interface found on AmigaLink, which should work perfectly, so I don't see what the problem could be, except for the software to be of a low standard.

Some new products have just surfaced, including a range of programs from Dr.T's. Their sequencer, K.C.S. (Keyboard Controlled Sequencer, about $495 from Brashs in the city) works well on the Atari so let's cross our fingers for the Amiga. They have also released a whole range of librarians and patch editors. Music-X is now out, I hear, as is Dynamic Studio, similar to Dynamic Drums I'm told, in that it is a combined drum machine (using the Amiga's sound channels and drum samples) and sequencer.

An Australian product, shown at one of our meetings at Burwood, called "Cquin" was meant to be interesting and not to expensive either. Yet it isn't available anywhere I've looked.

I hope these new products are quality. I'd like my computer to function satisfactorily in this department, as I have all the equipment but no decent software.

Amigalink has a music/midi message section so all you computer musicians with modems, or with access to

one, pull out your fingers, help stir up some interest and voice your problems. Somebody, even I, might know the answer. If any one has, or has used, any of the forementioned programs, leave a message with your detailed opinion about it! If anyone is running a midi studio with an Amiga as the centre of it then do the same, or better still write an article for this newsletter!

SMAUG (our midi/music special interest group) is meant to meet every month at the meeting, so I'll be there. I'm sure I'm not alone in my frustration, so come along next time and let's sort this problem out together. SMAUG is pretty dead at the moment yet things are happening, so let's get together. Bring, some gear with you. I'm thinking about it. Rock and Roll!

===

Lattice C V4.01 Review
By Andrew Conway

I have not used many C compilers, in particular I have not used AZTEC. Thus I am not perfectly qualified to write this review. Nevertheless, it may be interesting for people who are considering splurging on a C compiler.

Firstly some history. I started out using AmigaBasic on the Amiga, but soon got sick of interpreting and the messiness of interfacing to library routines. Thus I tried assembler. This was slow to program in, and it still wasn't very easy to use library routines. Finally the public domain C compiler on Fish Disk 110 came out. I eagerly tried it out. It worked! Well, it worked some of the time. It couldn't handle some syntax, floating point values, didn't have a pre-processor, and tended to produce 'guru'ing code. I often had to rewrite code several times until I got something that it liked.

It was then that I bought Lattice C. I was looking around for version 4, but couldn't find it. Eventually I purchased a copy of V3.10 on the assurance that Lattice would send me a free update to version 4. Meanwhile, I did a fair bit on 3.10. It was lovely - my code compiled quickly, reliably, and then only 'guru'ed if it was my mistake. I was very satisfied with version 3.10.

The update came fairly quickly and without fuss. It was complete, including a new reference book! I also get a quarterly "Lattice Works" newsletter, which is very interesting reading. I was very impressed with Lattice's support.

When version 4.01 arrived (totally free of charge), I tried it out immediately. The first thing I noticed was that the manual was in a better typeface, and 4 distribution disks meant that there must be a lot of extra stuff. However, the biggest difference was speed! The new linker (Blink 7.0) seems much faster that V6.7 (especially working from RAM:). The compiler also seems faster.

Version 4 has many special features. However, some of these are very poorly documented. The conversion of include files to packed include files is well documented, and probably makes a difference to speed of compilation (I haven't actually timed it). Also,

the fast floating point package is better integrated. These two features are well documented, as is directly calling library routines. Badly documented are the 16 bit integers option, and the part linking in Blink.

This bad documentation in these two places is my only complaint, however. It is not a problem to not use these features anyway. Overall, I highly recommend it, and I also recommend Lattice (who have also impressed me with support on other products). Do buy an original copy - don't just pirate someone else's. Doing so gets you the absolutely vital reference book, and good, ongoing support from Lattice.

AZTEC may be a better compiler, I don't know, but for almost all applications, I found Lattice's to be absolutely fine... and it is significantly cheaper.

[Editor's note: Am I the only person who hasn't received an upgrade from version 4.00 to V4.01? I still get my Lattice newsletters and everything, and I've upgraded (and payed my money!) every time Lattice has offered compiler upgrades. Perhaps I should drop them a line, but I'm interested to know if anyone else has been left out.]

===

To: Amiga Workbench Editor
From: Rudy Kohut

I was so moved by the editor's plea for material for the newsletter that I have decided to take the plunge, turn on the machine, load up my LPD Writer software and hey presto! As I am a new member, I have more questions to ask than advice to give. If anyone can help me with a few of these problems, I would be grateful.

First, why does my "CAPS LOCK" key repeatedly flash after booting (or re-booting the machine - an AMIGA 500)? This sometimes happens after using some public domain programmes such as "Overscan", which patches Intuition, if I don't remove the programme before closing down. It also happens sometimes (not all the time) after using some games. Because it does not appear consistently, I suspect that something is happening to Intuition, and that at boot, the startup is finding an inconsistency - although not enough to stop the boot. The machine works even with the key

flashing! However, a "soft boot" usually clears all. Do I have to be worried about this behaviour?

Second, I have found "Dmouse" an excellent public domain programme. It is a mouse blanker, screen blanker and CLI "pop screen" all rolled into one. However, one small bit of advice - DON'T use it with any paint programme that uses the mouse for drawing! With my "Graphics Studio" software, I get very strange results! In fact, the programme gets very confused and will crash - especially when trying to change screen resolutions. So be warned.

Third, my children use LOGO at school and I have obtained the version available through the public domain - a port of the Apple version. However, I find that there is enough inconsistency with the Apple version that using Apple based books is difficult for the kids. Is there a better version available for the Amiga? Any books based on the Amiga version?

Fourth, I'm in love with AmigaBasic, and the Abacus books, "AmigaBasic Inside and Out" and "Amiga Tips and Tricks", which are well worth getting hold of. Can anyone tell me if Microsoft/Commodore are going to release an upgrade of AmigaBasic? How can I find out (easily, that is!)?

Fifth, can anyone recommend a typing tutor for children aged 7-11? I have one called "Keyboard Cadet" which is very good for adults, but a bit too fast for children of that age. I don't want to teach touch typing, just good keyboard skills.

Last, some general comments. I'm very aware of the problem of trying to run a club that tries to please beginners, like myself, as well as the more "advanced" user. But the problem only becomes serious if the material for the advanced user is written in computer-ese. All communication is made better if we try to keep jargon to the minimum. For example, in my first query above I used the word "boot". It is so easy to forget that others may not know that this means "to start the machine"! It may take a bit longer to write out a phrase than a word, but then we can all participate in the discussion. Unless you know your audience, write as if you were trying to explain your ideas to a complete dunce! I'll bet we will find it hard to do!

I was interested in the debate about producing the club newsletter on an Amiga. My opinion is that the editor's comments are serious enough to bring to the attention of Commodore/Amiga. Surely, if we can't use the Amiga efficiently for our purposes with the software available, then Commodore should be very concerned. Let's put the situation to them and see what they suggest. I concur that the person doing the work should have the proper tools - and the newsletter is an excellent production as it is. But let's see if Commodore can come to the rescue. Worth a try?

Finally, as a Macintosh user at work, I appreciate the appeal of a machine that combines power with ease of use. That's why I bought this Amiga (price being a key factor as well!). So software that makes maximum use of both qualities is very important. Using the CLI is very interesting and a great extension of the user's ability to use the machine.

However, all that power should also be available through the Workbench environment. The key reason that the Mac is so successful in my workplace is that it is very powerful to use and no CLI-type commands are necessary. Here's hoping that version 1.3 improves things.

Thanks for your attention.

==

VideoTex, the Amiga and SuperTex
by John Morton

The VideoTex market place is growing rapidly and a local product which enables the use of the Amiga with VideoTex services is called SuperTex.

VideoTex data communications can be an exiting and rewarding extension of computer related pastimes. Telecom's Viatel is not the only VideoTex service available, and others have grown to meet the obvious market and user base. VideoTex is composed of screens of information or forms. The user controls the movement from screen to screen, either searching forward or backward through related screens or accessing a screen by its unique screen number. Related information is usually linked in this manner so its very simple to use. At the bottom of the screen there is usually an option line with the accepted codes and actions. Here are a couple of numbers in Victoria for VideoTex services,

```
Viatel     #01955       24hrs 1200/75 baud : Telecom
MouseTex   #(059)425528 24hrs 1200/75 baud : Anon
VTex 4000  #7413295     24hrs 1200/75 baud : Anon
```

The basic principles of VideoTex require the use of a modem , a monitor (preferably colour) and a keyboard terminal. Of course these components are part of the everyday computer system. This is where the Amiga comes in, using SuperTex (Version 2.0 - by Ron Wright & Peter Story) opens the world of VideoTex services to the home computer user. Australian services are growing rapidly and it is well worth the expenditure on good quality software like SuperTex. I use this product because it has been written especially for the Amiga, using Intuition Menu and Window structures. It is very easy to use and customize, a well written and informative manual accompanies release 2.0 of SuperTex.

For the past three months or so I have been travelling in Europe and I have seen a number of different implementations of VideoTex systems. In Great Britain for instance where all systems are at least in english VideoTex is available through your colour television by using the remote control as a keyboard to flick through each screen of information. The information provided is varied and usually very informative. Each morning I would wake and read the news headlines and in-depth articles directly from the television screen, check the weather forecast (which usually involved rain), and browse the programs available for the days watching. Both BBC1 & 2 had VideoTex and I think commercial channel 4. Much more information was available such as trading post, eating out, things to see and do, stock and trade figures. The system was available to everyone, and I found it extremely useful when planning a days outing.

In France the VideoTex services were part of your day to day existence, with each home having a MiniTel terminal. This fantastic little device with monochrome screen and keyboard was very compact, light to carry and plugged into a wall socket. It seemed where ever I went these terminals were being used. The connection was easy, just press the connect button on the terminal, there was no dialing required. Enter a numeric address with a password and in you were. All television stations offered services and broadcast their numeric address and password as public knowledge. There was an on-line telephone directory, where I was able to search for people or products by a number of methods. This service was very useful and hence the telephone book was not required. As you can imagine in a city like Paris, where more than 15 million people live, the telephone book would require constant updates and changes, and it would be gigantic, probably 2 trees per book (Save more trees!!). There were many other service providers offering specialized services and information databases all available through the home terminal. Also at all post offices there was a system called PostTel, which provided railway, bus, air, and shipping timetables, information and booking facilities. As a traveller this system was a valuable service and really added flexibility to travelling in a foreign country.

In Victoria the Viatel service offered by Telecom is gradually becoming more and more useful as new service providers are implementing extra services and facilities into the Viatel system. Viatel costs about $4.00 per month as a subscription for home use, and a connect charge of about 6 cents per minute plus the cost of a local call. Some screens have extra charges placed on them mainly because they deal with specific information relevant to the business and commerce sectors. At present I think Viatel is a little expensive and it still has a long way to go to match the systems in Europe. But as more people use and become familiar with VideoTex services hopefully some of these charges will decrease and some services will be offered free of any charges as exists in Europe.

To reduce my on-line charges I connect to Viatel and select each screen by screen number then save each screen in the carousel, which acts like a slide projector, saving each screen in memory. After I have finished saving all screens I want, I then disconnect. This reduces my on-line time considerably. I then save the carousel to disk in either Ascii or graphic mode and I can then read them at my leisure. SuperTex could do with a batching mechanism for this, but at present the screen and carousel save method is quite functional and fairly quick.

In summing up, I strongly recommend taking advantage of this growing information resource using your Amiga & SuperTex. VideoTex systems are used widely throughout the world and the quality and diversity of their uses can only increase.

==

World Darts
by Darren King

Hands up all of you reading this review who own or have owned a C-64! Do you remember a game called "180 DARTS"? It was a simulation where you competed against the computer or another player in a darts tournament. Well, darts lovers, a similar version has been released on the AMIGA and it is great!

World Darts is basically your normal game of darts. Starting from 501 points, you work your way back to 0 (which must be attained by getting a double of a particular score on the board). An announcer appears holding a microphone whenever it is your turn or you get "One Hundred and Eighty". He announces your luck in a very noticeable London accent that is really lifelike.

The game can be played by two human players, or a single player against the computer. You can select skill level (a very low skill level is often hard!), suggested shots, how many rounds per match and more. This game does take some getting used to in that your hand which throws the darts is a little shaky and takes some time in straightening up. All in all though, a great game if you like playing darts.

==

Potpourri by Allan Duncan

Peter keeps demanding articles for the newsletter, but he's shifting at the moment and hasn't got time to write all those you haven't sent in, so you're getting another one from me, whether you like it or not. The antidote is available to you all, just start writing your own.

Now for the material proper.

We all moan about the the American tendency to believe that the world ends at the Pacific and Atlantic shores. The Amiga has 200 active lines (400 in interlace) doesn't it? Well those of us over here and those Yanks using "morerows" know that it has more. All you need to set your maximum screen or window size to match the user's current value is available somewhere in the box. Where it is hiding can take a little ferreting out, but Carolyn Scheppner of Commodore Amiga Technical Support (CATS) has provided a handout to developers with an example of how to do it.

On a different, but ultimately allied, topic I wanted something other than the usual "CON:" or "RAW:" window for I/O purposes. One thing that annoys me about these is the limitations imposed by the default window gadgets that you get. For instance, the Window Sizing gadget "occupies" either a couple of columns or a row depending on whether the flag in the New Window structure is set for right or bottom margin. It doesn't matter which is used, there is always this penalty if a sizing gadget is used, and the CON/RAW window has it in the right hand border, so the maximum number of characters in a standard size window is 77 (two for the gadget and one for the border). I have looked around, but haven't found anything that will allow me to associate a window of my design with the CON: I/O task which actually works. There is a programme by Andy Finkel and Tim King on an early Fish disk that purports to do this, but doesn't work with 1.1 and 1.2 Kickstarts, and contains some non-portable high magic that taps into the guts of Exec, so I can't adapt it to current needs. I presume that it works under 1.0 - anybody got this to try it out?

The solution to my gripe is to go directly to the console.device, bypassing the filehandler stuff of CON:, associating my own window with the device and providing the necessary equivalents to getc() and putc(). The ROM Kernal Reference Manual - Libraries and Devices (aka RKM) has some examples of how to do this. The summation of these two streams is the sample programme in the vicinity of this piece. Any suggestions on better ways, goofs, etc are welcome, both here and on AmigaLink. For those who want to play with the code but are too lazy to retype it, I'll post it in the C files area.

While I'm rambling on, there are a couple of other things that may be of interest. Those that have read this far are probably programmers, and will be aware that many system function calls return pointers to structures that contain information beyond your wildest dreams. Avid readers of other people's programmes (particularly those from CATS)

will have noticed a few discrepancies between the advertised return values and the usage that is made of it. A case in point is the return from GetMsg() in my code. The book says that it returns a pointer to a Message structure, but some of the things you need to do involve members of the IntuiMessage structure, which has the Message structure as its first element. Nowhere can I find it stated formally in the RKM that the Message that GetMsg() returns is the one at the head of the IntuiMessage, but all the examples given make this assumption, and since the people that wrote these know what Exec actually does, it seems that this is a safe bet. There seems to be a certain cavalier attitude by those with the earlier Lattice compiler, where a pointer is a pointer is a pointer, and to Hell with casts and function prototyping that might force care and responsibility onto pointer use. Compare the following sample from the RKM with the sanitized version in ConGetChar() of my code.

```
struct IntuiMessage *msg ;
msg = (struct IntuiMessage *)GetMsg(...) ;
class = msg->Class ;
ReplyMsg( msg ) ;
```

You can get away without the cast if you declare GetMsg() as returning a pointer to IntuiMessage. Note that the argument to ReplyMsg() should be a pointer to a Message.

In a similar vein, the return from a FindTask() is a pointer to a Task structure, and examination of the include files shows that this is contained within a Process structure as the first element. Sure enough, code fragments indicate that the Task returned is the one in a Process. Not only that, there are indications that the next element, a MsgPort, is returned by DeviceProc() or Create() and the appropriate Process can be obtained by

```
(struct Process *)( DeviceProc(...) - sizeof(struct Task) )
```

To wind up, a comment on my code, or rather what is not there. While I was putting it together, the need to keep track of resources that I have obtained (successfully or not) was a pain. If you want to build and test as you go along, you end up working in three spots at once - the startup part where the resources are obtained, the functions that are going to do what the programme is for, and the cleanup code to deallocate in reverse order all that, and only that, which you still own. In a large programme this is a big problem. It dawned on me while reading an item on linked lists that these would be the way to go. Set up a list structure that holds a pointer to the previous instance, a pointer to a function that will release whatever it is, and a pointer to some argument or array of pointers to arguments to be used by this function. Come home time, just take the tail pointer and traverse the linked list in reverse order, executing the functions that are associated with a non-NULL argument pointer. If portions of code need to be able to release a resource during normal operation, they can keep their own references to the list items in question, and clear the argument pointer to flag the completed release. Obviously there needs to be some allocation of memory space for each instance of the list structure, and provision for removal at the end, but it is quite feasible. This didn't make it to this little programme, and is "left to the reader".

```
/*     win.c   full screen size window test */
/* A collation and adaptation by Allan Duncan of examples in the RKM Console.Device
      Chapter and from Carolyn Scheppner of CATS */

#include      exec/types.h
#include      intuition/intuition.h
#include      functions.h
                         /* Pre-defined function specifications */
#include      arpfunctions.h
                         /* ditto */

#ifdef  NULL
#undef  NULL
#endif
#define  NULL  0L     /* Manx wants Longs for talking to Amiga functions */

int      ConGetchar() ;
struct  IntuitionBase   *IBase ;
struct  Window          *myWindow ;
      /* These are the various things for the window we are creating */
struct  NewWindow nw=(
      0, 0,            /* SHORT LeftEdge, TopEdge */
      0, 0,            /* SHORT Width, Height */
      -1, -1,          /* UBYTE DetailPen, BlockPen */
      CLOSEWINDOW,     /* ULONG IDCMPFlags Notify us of this */
                       /* Gadgets we want :- */
      ACTIVATE | WINDOWCLOSE | WINDOWDEPTH | BORDERLESS,
      0,               /* pointer for extra gadgets */
      0,               /* pointer for CheckMark, use system's default */
      (UBYTE *)"Test Window",    /* Title for our window */
      0, 0,            /* For custom Screen/BitMap if we wanted */
      0, 0, 0, 0,      /* Resizing limits if we had sizing gadget */
      WBENCHSCREEN     /* USHORT Type - ... or CUSTOMSCREEN etc */
      ) ;

struct  MsgPort           *rport=NULL, *wport=NULL ;
struct  IOStdReq          *rdIOreq=NULL, *wrIOreq=NULL ;
UBYTE inchar ; /* storage for use by the reading task */
int      tempchar ;
int      flagCons=0 ;

main()
{
struct  Screen  wbScrData ;
      /* First we want to find the user's current WB Screen dimensions ... */
if ( !(IBase = (struct IntuitionBase *)OpenLibrary("intuition.library",33L)))
                       cleanup("Can't open intuition") ;
      /* which will be copied into our wbScrData by this :- */
if ( !(GetScreenData(&wbScrData, (long )sizeof(struct Screen), WBENCHSCREEN,
            NULL))) cleanup("Can't get Workbench screen info") ;
nw.Width = wbScrData.Width ;   /* Put this into our NewWindow structure,... */
nw.Height = wbScrData.Height ;
nw.TopEdge = nw.LeftEdge = 0 ;
      /* and open ! */
if ( !(myWindow = OpenWindow( &nw ) ) ) cleanup( "At least V1.2 KS required" ) ;
      /* Create a series of RastPorts for communication with it */
rport = CreatePort( "myreadport", 0L) ;
if ( rport == NULL ) cleanup("Can't create Read port") ;
rdIOreq = CreateStdIO( rport ) ;
if ( rdIOreq == NULL ) cleanup("Can't create Read message") ;
wport = CreatePort( "mywriteport", 0L) ;
if ( wport == NULL ) cleanup("Can't create Write port") ;
wrIOreq = CreateStdIO( wport ) ;
if ( wrIOreq == NULL ) cleanup("Can't create Write message") ;
      /* Get a console.device for ANSI 3.64 character i/o,
            try using your cursor keys, or specials like esc[4m */
if (OpenConsole(wrIOreq, rdIOreq, myWindow)) cleanup("Couldn't open console") ;
flagCons = 1 ;nw.Width ; /* Note that we've managed to open the console */
QueueRead(rdIOreq, &inchar) ;   /* Put the read task into operation */

      /* Put your code from here ..... */
ConPuts(wrIOreq, "Enter your string here -") ;
while ( (tempchar = ConGetchar(rport, rdIOreq, &inchar) != -1 )
                            ConPuto( wrIOreq, (UBYTE )tempchar) ;

      /* .... to here */

cleanup("") ;
}

cleanup(errormsg)      /* Close up all that (and only that) we have opened */
char    *errormsg ;
{
struct  Message           *msg ;  /* This was IntuiMessage in Carolyn's example */

if ( flagCons ) CloseDevice(wrIOreq) ;
if ( rdIOreq ) DeleteStdIO(rdIOreq) ;
if ( wrIOreq ) DeleteStdIO(wrIOreq) ;
if ( rport ) DeletePort(rport) ;
if ( wport ) DeletePort(wport) ;
if( myWindow ) {
      while ( msg= GetMsg(myWindow->UserPort) ) ReplyMsg(msg) ;
      CloseWindow(myWindow) ;
      }
if ( IBase ) CloseLibrary(IBase) ;
if ( *errormsg == '0' ) exit(0) ;
Printf("%s",errormsg) ;
exit (20) ;
}

OpenConsole(wrIOreq, rdIOreq, wind)
struct  IOStdReq       *wrIOreq, *rdIOreq ;
struct  Window         *wind ;
{
int      error ;

wrIOreq->io_Data = (APTR )wind ;
wrIOreq->io_Length = (ULONG )sizeof(*wind) ;
error = OpenDevice("console.device", 0L, wrIOreq, 0L) ;
rdIOreq->io_Device = wrIOreq->io_Device ;   /* Clone required parts */
rdIOreq->io_Unit = wrIOreq->io_Unit ;
return (error) ;
}

ConPutc(IOrequ, chr)
struct  IOStdReq       *IOrequ ;
UBYTE chr ;
{
IOrequ->io_Command = CMD_WRITE ;
IOrequ->io_Data = (APTR )&chr ;
IOrequ->io_Length = (ULONG )1L ;
DoIO(IOrequ) ;
}

ConPuts(IOrequ, str)
struct  IOStdReq       *IOrequ ;
UBYTE *str ;
{
IOrequ->io_Command = CMD_WRITE ;
IOrequ->io_Data = (APTR )str ;
IOrequ->io_Length = (ULONG )(-1L) ;   /* null terminated string */
DoIO(IOrequ) ;
}
      /* Set up a request for a character from the read stream, which will
         be parked in "whereto" ready for us when we get round to looking */
QueueRead(IOrequ, whereto)
struct  IOStdReq       *IOrequ ;
UBYTE *whereto ;
{
IOrequ->io_Command = CMD_READ ;
IOrequ->io_Data = (APTR )whereto ;
IOrequ->io_Length = (ULONG )1L ;
SendIO(IOrequ) ;
}

      /* Actually fetch the waiting character, or wait if there is none. Also */
ConGetchar(conPort, IOrequ, whereto)   /* wait on the CloseWindow gadget */
struct  IOStdReq       *IOrequ ;
struct  MsgPort        *conPort ;
UBYTE *whereto ;
{
register int      temp ;
struct  Message *msg ; /* This was IntuiMessage in Carolyn's example */
long    sigret, signal1, signal2, class ;

temp = 0 ;         /* used for the function return value */
signal1 = 1L << conPort->mp_SigBit ;   /* for the character ready */
signal2 = 1L << myWindow->UserPort->mp_SigBit ; /* for the CloseWindow */
      /* Do we have a waiting char ? */
if ( (msg = GetMsg(conPort) ) ) {  /* Don't ReplyMsg these, their ours */
      temp = *whereto ;          /* get the character and recycle the request */
      QueueRead(IOrequ, whereto) ;
      return temp ;
      }
sigret = Wait( signal1 | signal2 ) ; /* No, so sleep until one of our events happens */
      /* Got something, is it a character ? */
if ( sigret & signal1 && (msg = GetMsg(conPort)) ) {
      temp = *whereto ;
      QueueRead(IOrequ, whereto) ;
      }
      /* It SHOULD be a CloseWindow */
if ( (sigret & signal2) && (msg = GetMsg(myWindow->UserPort)) ) {
      class = ((struct IntuiMessage *)msg)->Class ;   /* What sort was it ? */
      ReplyMsg(msg) ;          /* This wasn't ours, so send it back */
      if ( class == CLOSEWINDOW ) {
            temp = -1 ;    /* Flag it like an EOF */
            }
      }
return temp ;   /* It might be that 0 if the above went wrong somehow */
}
```

Camberwell Computer Fair
by Lester McClure

On Sunday 18th of September, the day of last months' AUG meeting, a 'Computer Fair' was held at the Camberwell Civic Centre. It was billed by the commercial organisers as a new concept, the first of its kind in Melbourne, with a NEW section for retailers and a USED section for individuals.

Computer clubs were also invited to participate and our committee made a last-minute decision to attend and represent the Amiga users community. Demonstrations disks were put together along with some 'flashy' games, the club's Amiga 500 and a few photocopied handouts. It was difficult to get help from others because of the clash with our monthly meeting, so thank you Bob, John, Craig and Michael for your assistance.

The fair ran from 10 am until 6 pm, and was quite small in comparison with shows like PC88, but I liked the smaller format and the greater diversity, due mainly to the USED and club exhibits. It was disappointing that the Amiga was not being actively demonstrated by dealers, although one was showing the 'Amiga Movie' and another had a limited range of software for sale. I believe that Commodore really missed a great chance to promote the Amiga, particularly the domestic A500 pack.

The organisers had scheduled short talks throughout the day on various topics but I was unable to attend any of them since I spent most of my time talking to many people interested in the Amiga and our Users Group. They ranged from family groups that had just bought an Amiga package because it seemed like a good idea, to people who knew quite a lot about Amigas but little about our club. The prize has to go to the rotund chap who rolled up and stunned me with "Well, what computers can the Amiga emulate then?", he had obviously come straight from the Atari users group where they were demonstrating a machine that wasn't sure if it was IBM PC, a MAC clone or a worthwhile computer system in its own right.

Overall the day was quite a success, two new members joined AUG and several others have returned our special membership form since. More importantly though, was the general comment from so many people that we would have been noticed in our absence - who else is prepared to support the Amiga if we don't? Next year (I'm sure the fair will become an annual event) we should hire a bigger space and encourage more members to spend some time displaying a greater range of machines/demos, not everyone had a chance to see what they wanted.

==

Importing Software Direct From The USA
By Peter Ward

I feel that this may be a little like describing to people how to "suck eggs", but a new service described in the September edition of "Amiga World" really does deserve some attention from us Australian Amiga users. As an example: Word perfect will cost you about $350 Australian including tax duty and postage, not bad eh?

Lightspeed Distributions now have a International Toll-Free number for us here in Australia. The number is: 0014-800-12-5632. Call between the hours of 7am to 6pm Monday to Friday or 9am to 3pm Saturday, West Coast time for the USA (or before 10am here in Oz if you can't be bothered with the time conversion). Prices are good, service is prompt. My most recent order arrived in 14 days. All you need is a telephone and either a Visa or Mastercard number, dial said number and spend all the time you want giving your order or enquiring about the product then give card details without that nagging feeling this is costing you about a dollar a sentence. There is a catch, you have to buy more than US$100.00 worth of software for use of the toll free service.

I realize that I may get some criticism for suggesting the use of a foreign retailer, and not supporting the local distributors, but I think there is a difference between reasonable profit and out-right ripoffs, so perhaps the use of services such as this might help toward a drop in local prices.

[Editor's note: Potential purchasers of software and/or hardware might also note that most Australian distributors do not provide support for the packages they distribute at their highly inflated prices. Since some local companies do do the right thing, however, you might like to ask around at an AUG meeting about local support for the package you want to buy before deciding to buy locally or direct from the USA.]

==

UES = Uedit
by Lachlan Myers

I don't like shareware. The whole idea stinks of exploitation, since the software relies on the PD software sharing system for its distribution. These arguments are put rather well by Steve Leon, who ran the CP/M SIG/M library for many years (remember CP/M? - still a better file-server than MS-DOS 3 etc, and, in its Z-System variants, a bloody good one). He says:

"What we are trying to do is make available Public domain software, including non-CP/M material, that is not sitting there as beggar-ware, asking for a donation. What we seek is good quality material, preferably with source code. It cannot be a demo of commercial software or make any request for a contribution."

(Micro/Systems Journal, Vol.2 No.6 Nov/Dec 1986 p76)

However, I needed a good editor - with wordprocessing capability if possible.

I tried DME, tried ED, tried E, then went back to the CP/M machine for Wordstar 4, still the first and best, and piped files back at 19200 baud after print to disk. But that kind of performance gets very wearing, and the Fish disks showed an editor called Uedit (vol 121), which I obtained and gradually worked my way around. This was version 2.3c, and I later found version 2.3h on the (AmigaLink) BBS. The latest version seems much more solid [I had a few screaming fits with the earlier one] and integrates

---

Intro to Amiga IFF ILBM Files and Amiga Viewmodes
by Carolyn Scheppner
Commodore Amiga Technical Support

The IFF (Interchange File Format) for graphic images on the Amiga is called FORM ILBM (InterLeaved BitMap). It follows a standard parsable IFF format.

Sample hex dump of beginning of an ILBM

Important note! You can NOT ever depend on any particular ILBM chunk being at any particular offset into the file! IFF files are composed, in their simplest form, of chunks within a FORM. Each chunk starts starts with a 4-letter chunkID, followed by a 32-bit length of the rest of the chunk. You PARSE IFF files, skipping past unneeded or unknown chunks by seeking their length (+1 if odd length) to the next 4-letter chunkID.

```
0000: 464F524D 00016418 494C424D 424D4844   FORM..d.ILBMBMHD
0010: 00000014 01400190 00000000 06000100   .....@..........
0020: 00000A0B 01400190 43414D47 00000004   .....@..CAMG....
0030: 00000804 434D4150 00000030 001000E0   ....CMAP...O....
0040: E0E00000 20000050 30303050 50500030   .... ..P000PPP.0
0050: 90805040 70707010 60E02060 E06080D0   ..P@ppp.`.`.`..
0060: A0A0A0A0 90E0C0C0 C0D0A0E0 424F4459   ...........BODY
0070: 000163AC F8000F80 148A5544 2ABDEFFF   ..c.......UD*...
0080: FFBFF800 0F7FF7FC FF04F85A 77AD5DFE   ...........Zw.].
etc.
```

Interpretation:

```
       'F O R M' length 'I L B M''B M H D'<-start of BitMapHeader chunk
0000: 464F524D 00016418 494C424D 424D4844    FORM..d.ILBMBMHD

      length  WideHigh XorgYorg P1MkCoPd  <- Planes Mask Compression Pad
0010: 00000014 01400190 00000000 06000100    .....@..........

      TranAspt PagwPagh 'C A M G' length  <- start of C-AMiGa View modes chunk
0020: 00000A0B 01400190 43414D47 00000004    .....@..CAMG....

      Viewmode 'C M A P' length   R g b R  <- Viewmode 800=HAM | 4=LACE
0030: 00000804 434D4150 00000030 001000E0    ....CMAP...O....

       g b R g  b R g b  R g b R  g b R g  <- Rgb's are for reg0 thru regN
0040: E0E00000 20000050 30303050 50500030    .... ..P000PPP.0

       b R g b  R g b R  g b R g  b R g b
0050: 90805040 70707010 60E02060 E06080D0    ..P@ppp.`.`.`..

       R g b R  g b R g  b R g b 'B O D Y'
0060: A0A0A0A0 90E0C0C0 C0D0A000 424F4459    ...........BODY

      length   start of body data   <- Compacted (Compression=1 above)
0070: 000163AC F8000F80 148A5544 2ABDEFFF    ..c.......UD*...
0080: FFBFF800 0F7FF7FC FF04F85A 77AD5DFE    ...........Zw.].
etc.
```

Notes on CAMG Viewmodes: HIRES = 0x8000
                            LACE = 0x4
                            HAM = 0x800
                         HALFBRITE = 0x80

BMHD - info about the size, depth, compaction method (See interpreted hex dump at left)

CAMG - optional Amiga viewmodes chunk. Most HAM and HALFBRITE ILBMs should have this chunk. If no CAMG chunk is present, and image is 6 planes deep, assume HAM and you'll probably be right. Some Amiga viewmodes flags are HIRES=0x8000, LACE=0x4, HAM=0x800, HALFBRITE=0x80.

CMAP - RGB values for color registers 0 to n (each component left justified in a byte)

BODY - The pixel data, stored in an interleaved fashion as follows: (each line individually compacted if BMHD Compression = 1)

```
plane 0 scan line 0
plane 1 scan line 0
plane 2 scan line 0
...
plane n scan line 0
plane 0 scan line 1
plane 1 scan line 1
etc.
```

Body Compression

The BODY contains pixel data for the image. Width, Height, and depth (Planes) is specified in the BMHD.

If the BMHD Compression byte is 0, then the scan line data is not compressed. If Compression = 1, then each scan line is individually compressed as follows:

More than 2 bytes the same stored as BYTE code value n from -1 to -127 followed by byte to be repeated (-n) + 1 times.

Varied bytes stored as BYTE code n from 0 to 127 followed by n+1 bytes of data.

The byte code -128 is a NOP.

ILBM is a fairly simple IFF FORM. All you really need to deal with to extract the image are the following chunks:

(Note - Also watch for AUTH Author chunks and (c) Copyright chunks and preserve any copyright information if you rewrite the ILBM)

Interpreting the Scan Line Data

If the ILBM is not HAM or HALFBRITE, then after parsing and uncompacting if necessary, you will have N planes of pixel data. Color register used for each pixel is specified by looking at each pixel thru the planes.

IE - if you have 5 planes, and the bit for a particular pixel is set in planes 0 and 3:

```
PLANE    4 3 2 1 0
PIXEL    0 1 0 0 1
```

then that pixel uses color register binary 01001 = 9

The RGB value for each color register is stored in the CMAP chunk of the ILBM, starting with register 0, with each register's RGB value stored as one byte of R, one byte G, and one byte of B, with each component left justified in the byte. (ie. Amiga R, G, and B components are each stored in the high nibble of a byte)

BUT - if the picture is HAM or HALFBRITE, it is interpreted differently.

Hopefully, if the picture is HAM or HALFBRITE, the package that saved it properly saved a CAMG chunk (look at a hex dump of your file with ascii interpretation - you will see the chunks - they all start with a 4-ascii-char chunk ID). If the picture is 6 planes deep and has no CAMG chunk, it is probably HAM. If you see a CAMG chunk, the "CAMG" is followed by the 32-bit chunk length, and then the 32-bit Amiga Viewmode flags.

HAM pics will have the 0x800 bit set in CAMG chunk ViewModes. HALBRITE pics will have the 0x80 bit set.

To transport a HAM or HALFBRITE picture to another machine, you must understand how HAM and HALFBRITE work on the Amiga.

How Amiga HAM mode works

Amiga HAM (Hold and Modify) mode lets the Amiga display all 4096 RGB values. In HAM mode, the bits in the two last planes describe an R G or B modification to the color of the previous pixel on the line to create the color of the current pixel. So a 6-plane HAM picture has 4 planes for specifying absolute color pixels giving up to 16 absolute colors which would be specified in the ILBM CMAP chunk. The bits in the last two planes are color modification bits which cause the Amiga, in HAM mode, to take the RGB value of the previous pixel (Hold and), substitute the 4 bits in planes 0-3 for the previous color's R G or B component (Modify) and display the result for the current pixel. The color modification bits in the last two planes are interpreted as follows:

00 - no modification. Use planes 0-3 as normal color register index

10 - hold previous, replacing Blue component with bits from planes 0-3
01 - hold previous, replacing Red component with bits from planes 0-3
11 - hold previous. replacing Green component with bits from planes 0-3

How Amiga HALFBRITE mode works

This one is simpler. In HALFBRITE mode, the Amiga interprets the bit in the last plane as HALFBRITE modification. The bits in the other planes are treated as normal color register numbers (RGB values for each color register is specified in the CMAP chunk). If the bit in the last plane is set (1), then that pixel is displayed at half brightness. This can provide up to 64 absolute colors.

Other Notes

Amiga ILBMs images must be a even number of bytes wide. Smaller images (such as brushes) are padded to an even byte width.

ILBMs created with Electronic Arts IBM and Amiga "DPaintII" packages are compatible (though you may have to use a '.lbm' filename extension on an IBM). The ILBM graphic files may be transferred between the machines (or between the Amiga and IBM sides your Amiga if you have a CBM Bridgeboard card installed) and loaded into either package.

===

Latest Public Domain Disks

Only two disks listed this month, although thirty-three disks arrived in. The other disks are from collections that are not catalogued in any way, so someone has to physically run every program and read every text file on each disk to document them fully. As this is done, these contents of these disks will be listed and they will be available from our disk library.

Amigan Disk #15

Iff2Ps	Useful shareware program to convert IFF pictures to PostScript files, for use on laser printers.
MR	Menu Runner - a menu bar clock, dynamic free memory reporter, and program execution utility all rolled into one.
ClickToFront	A double click into any window instantly brings it to the front; no more hunting for page gadgets. If you work with lots of windows, this one saves time.
DefDisk	Handy program that will remedy the need to use the DOS Assign command repetitiously in your Startup-Sequence to reassign the six standard assignments (SYS:, C:, etc.) to another path such as a hard disk. With source.
DOSKwik	Makes an image of an entire ramdisk (or survivable ramdisk) in one gulp on disk--and will copy that image back to ramdisk (complete with all directories) in another gulp. Over

twice as fast as COPY ALL. Great for fast boots--or for saving ramdisk entire at 2 a.m. in the morning so you can load it again when sober.

NewCon	A handy little utility to provide 'keyclick' audible feedback from the Amiga (provided you have your speakers hooked up). Also has a screen blanking function (who doesn't, anymore?)
ScreenSizer	A utility to allow you extra rows and columns of text in a CLI window. Provides all functions that 'MoreRows' does and also makes the current (usually Workbench) screen realize the changes without rebooting. Includes source.
SuperView	The best picture viewer so far? You decide. Will show almost ALL current picture types on the Amiga, including Overscan, the first frame from an ANIM file, and even AmigaBASIC .ACBM pictures!
RunBack	Will run a utility in the background, allowing you to close the CLI window. Handy when you don't want to clutter the screen with unneeded windows (or tie up a CLI which really isn't needed. Not needed if a program has been created to automatically run in the background--but many haven't.
APL.util	A series of utilities for those who use Spencer APL on Amiga. Covers basic functions.
Keep	For telecommunication junkies. You can review downloaded files off-line and save only those you want.

Amigan Disk #16

Arp1.1	The OFFICIAL Release 2 of the AmigaDOS Replacement Project programs. Accept no substitutes!
ConMan1.1	This is a SIGNIFICANT update to the release on Amigan Disk #11. (Hey, ABasiC works again!)
FixVDK	Patches The Survivor, otherwise known as VDK:
Flip1.1	Sideways printing program. Works with any Epson compatible printer hooked up to the parallel port.
Iff2Pcs	An interesting diversion, as well as good example code for several areas. (Reading of IFF, graphics manipulation, etc.)
LDebugDemo	Well, MANX has one (maybe), so LogiComp did it for Lattice. This is a demo version of a Source Level Debugger for Lattice C and Assembler.
Mach	A program which does a reasonable job of doing EVERYTHING! Mouse speedup, clock, PopCLI, SUN/HeliosMouse, Click-to-Front, Screen-Blanker, Hotkeys, etc, etc.
Patch1.2	EVERYBODY needs to use this file!!! Patches two bugs in Kickstart 1.2 that can cause random visits from the GURU!
VGad	A PD PowerWindows. Not as easy to use, but the price is certainly right!

Undelete	It will undelete a file you accidentally destroyed, so long as it isn't overwritten.
Chatty	The source code and executable for John Toebes' demonstration program, Chatty--which shows how to store messages and text in an .info file. These can be in any language (French, German, English) and revised at will without recompiling a program. See John's Column, "Oh Say Can You C," which is on this disk in the file CHATTY.DOC. The Chatty directory will show no drawer icon on WB because the chatty.info file within the Chatty directory might be confused with the drawer file. No problem. This stuff is for advanced C programmers.

Amiga Transactor Disks

Also available from our disk library are the disks for **Transactor for the Amiga**, issues 1 and 2. These disks are copyright, but are available for purchasers of **Transactor for the Amiga** magazine. Here is a listing of the disk for issue 1, along with a listing of the first Amiga Transactor PD disk.

Transactor Disk for Vol 1 Issue 1

TextReader	A file reader that you can use from CLI or Workbench, with proportional Gadgets for vertical and horizontal scrolling, and Gadgets for scrolling up and down through the text line-by-line.
Arc22	Some of the directories contain archived sets of files, which must be decomposed using Raymond S. Brand's shareware ARC program. This is version 0.22, which is not the latest, but will work on all the files here.
Arp	This directory contains ALL the materials for the AmigaDOS Replacement Programs project, which includes CLI command replacements that give you shorter, faster and more powerful equivalents for virtually all the standard CLI commands.
Cycles	The server program and players for Rico Mariani's multiple-task Light Cycles game from the magazine. Source code and executables are included. If you decide to write a player yourself (see the magazine article for how to do that), please share it with us.
DosPackets	The example program source and executables for Matt Dillon's tutorial article on programming with DOS Packets in C.
FixHunk	The FixHunk program recommended in Bryce Nesbitt's Amiga Programming Errors article.
JBEcho	The assembler source and executable for Jim Butterfield's version of Neal

JTCalc
　　Bridges' version of the CLI Echo command, with support for console escape sequences and other options. The sources and executables for the first and final versions of the calculator programs discussed in John Toebes' tutorial on porting C programs from Unix(tm) to the Amiga.

Lists
　　Source and executable for Rob Peck's tutorial article on using Exec Lists from C.

MemWatch
　　The MemWatch program recommended in Bryce Nesbitt's Amiga Programming Errors article.

MoreRows
　　The MoreRows program recommended in Bryce Nesbitt's Amiga Programming Errors article.

SB1.3
　　The latest version of TransAmi's Structure Browser utility, which will let you look at the current contents of a wide variety of system data structures, especially those relating to Intuition and the Graphics library. To use Structure Browser, enter:

　　　　　run sb

from the CLI, then follow your mouse. Structure Browser attempts to protect you from address and other exceptions generated by referencing from pointers contained in structures that get altered or dismantled while the program is running, but avoid attempting to examine structures that you know no longer exist, ok? Complete C source is included.

VirusProtect
　　The programs described in J.C. Bat's article on the SCA Amiga Virus, including the updated version of VCheck.

Transactor Amiga PD Disk #1

This disk consists of works by various authors. Some of the programs and articles have appeared in the Transactor, and others are public-domain programs that we have included because of their significance.

An exception is "Uedit", an outstanding text editor. This program is shareware, and it does not come free on this disk as part of the purchase price; we are including it in case you don't have access to it from another source (or want to save the time and money downloading it from an on-line service or BBS). If you find UE as useful as we do (very), please send the author the money and become a registered user (which can make YOU some cash - see the ReadMe file in the UE directory).

The following is a brief overview of the programs on this disk. For more information on any of these, double-click the ".doc" icon in its drawer.

Free Utilities:

BLink
　　A linker for Lattice C or Amiga assembler that is faster and has more features than ALink.

Asm68K
　　A full-blown 68000 assembler! Use it with BLink and build yourself a FREE

program development system.

Csh
　　A super CLI-replacement DOS shell by Matt Dillon. Supports editing of previous command history, filename wildcard expansion, aliases, built-in commands, source-file capability with IF ELSE, etc., and more. This version (V2.05) with modifications by Steve Drew.

ARC
　　A powerful file archiver: compresses many files together into one file that can be uploaded or downloaded easily. Lets you add, extract, or list files in an archive, and more. Supports three kinds of file compression, and uses the most efficient method for each file. ARC is freely distributable, but the author accepts contributions for his considerable efforts. Commercial or government users of ARC must pay a $35.00 licensing fee. See the doc file in the ARC directory.

Asm6502
　　Here's a powerful 6502 cross assembler/linker for the Amiga. Develop your 6502 programs on a high-powered Amiga environment using a linker, then send the final code to the target machine for testing.

ShareWare Utilities:

Uedit
　　A totally programmable, full-featured text editor. This program is shareware (see note above).

Transactor Programs:
(These are also freely distributable)

SB
　　The famous Transactor Structure Browser, expanded from the original. SB Lets you view system data structures in currently executing programs and branch to new structures via pointers. This version supports many Intuition structures and some graphics ones as well. Does hex and text dumps where appropriate. All browsing is done just by clicking with the mouse.

PopToFront
　　Brings the currently active window to the front when Left-Amiga F is pressed. Lets you pop a window to the front with a click and a keypress instead of a lot of window-shuffling! From The Transactor, Volume 8 Issue 1.

TrapSnap
　　Lets you easily add a trap-handler to your C programs to avoid the "Software Error" requester when a CPU exception occurs. From The Transactor, Volume 8 Issue 1.

PopColours
　　Lets you set the Red, Green and Blue components of any colour register with proportional gadgets on ANY SCREEN in the system. You can use it to modify the colours of any program that is currently running. When the PopColours window is inactive, it displays only a title bar, which pops to the full control panel when

activated. Also included is PopColours 1.2, which uses an on/off toggle switch to switch between small and large windows.

XE
　　An expression-evaluator calculator. Add, subtract, multiply and divide using standard mathematical expressions and variables. Works in any number base.

Function Plot
　　An AmigaBasic program that plots any function, automatically scaling the graph to any window size.

===

Editor's Column
(Written 01-Oct-88)

Boy, has this newsletter been a hassle! I've moved house, for those of you who haven't heard. When I say moved, perhaps I'm giving the wrong impression - I'm *moving*, actually. Almost everything, including the computer I'm typing on, has been moved into our new house, there's really only cleaning up to do at our old abode. The main principle involved in moving seems to be to put everything from house A into little boxes, put them in the moving van, take them to house B, then spend the next four years trying to find things again!

So, in the midst of all that, there's a newsletter to do. Lucky, aren't I....

I seem to have found the computers, printers, paper and the two articles that members submitted on disk. This, unfortunately, does not a newsletter make. Most of the newsletter articles this month were sent to me via AmigaLink, our BBS, which is a great idea. That way, I didn't have to work out which box I packed them into!

We had quite a talk about the newsletter and the general lack of articles at our main meeting last month, so I won't bore you with that topic again, except to say that I'm disappointed to realise that most members do not want to contribute to the club unless they are paid money to do it.

To finish my column this month, I'd like to finish my column - I am tendering my resignation as editor. This is my twenty-ninth newsletter, and I shudder when I realise how much time I've put into the group in general, and into being editor in particular. Most months, I only have one weekend free of AUG business in some shape or form.

So, I've decided that there are other things in life besides newsletters, and after two and a half years, I need a break. I don't intend leaving the group in the lurch, and I'm quite prepared to help our new editor "settle in" for a reasonable time.

The biggest problem is that there is no "heir apparent" - no would-be editor waiting in the wings for my departure. Someone will have to do just what I did when I started the group - jump right in with both feet. Perhaps a different personality might be able to persuade the general membership to write more articles.

So long, and thanks for all the fish.

Games SIG Report
by Luke Devlin

Apart from the pandemonium experienced at the September edition of the Games SIG, we showed Carrier Command, P.O.W, AAARGH, Star Ray pre-release, The Empire Strikes Back, and Starglider II.

We hope to be a lot more organised at the next meeting with the return of Anthony who will be just off the plane from Hong Kong, hopefully with all the latest (Original) games.

The crowd was amazed at my proposal that I, the organizer of the games SIG, was going to try to produce a program for the Amiga. If there are any programmers out there who would like to help me, I will be glad to give them a share of the profits for their contributions. Maybe we can make the program a users group venture? The hardware is already under construction and you will probably see the finished prototype after I finish my exams!

At the meeting, I showed everyone what you should do with disks that have hard errors and proceeded to break them open and use them as Frisbees. Two days after this, I found out that you could rectify Hard errors with Bformat, such is life I suppose. I also found out that in some cases DiskDoctor will undelete files and there is another program called UNdelete which will revive old deleted files.

I have purchased a modem and as soon as it is delivered, there will be weekly Games SIG messages left on our BBS. Then I will be able to get some more great ideas on how to improve our meetings.

The SIG also helped some people with answers to those questions on games, parts that they couldn't get through, where you should purchase, the best price?, is it better to purchase from the states? and so on.

At the October meeting we will be showing Quix, Bomb Jack, Foundations Waste, Better dead than Alien, SurcoPhasar, Bo Bo, Future Tank and all the other new programs that people care to bring along. PLEASE bring all your new games to show the world!

I'm hoping to see YOU there! Let's make this meeting a real organised and good meeting and I want to see LOTS of dedicated gamers show up. You all must have enjoyed playing at least one of the Amiga's great games, so come along and see all the latest ones and tell us what you think the all-time-greatest game is.

===

Publicity Officer's Report
by Luke Devlin

To those who don't know me, I'm the latest addition to the committee. I've been appointed as Publicity Officer to promote the Users Group and to try to increase membership and the participation rate. At the moment I'm undertaking H.S.C (Studying Business subjects) and have a very keen interest in the Amiga.

I'm 17 years old, and although young I hope to make up for that in enthusiasm. I do have some experience as I have owned my amiga for 2 years, on holidays I have been employed as Myer Chadstone's Amiga "Expert" and managed to sell an Amiga every day, and I also run the Games SIG with a little help from others.

Although I'm not really an expert on the Amiga, I was the one with the most knowledge about it at Myer. Compared to many in the Users Group, I'm but a toddler in the world of programming.

The one thing that people will find out about me is that once you get me revved up you can't stop me! I really like our Users Group and I want to see it grow and become great instead of fold due to lack of interest.

The first thing I would like to see at the meetings is MORE people. We have about 1000 members, yet I only counted around 170 at the September meeting. Where are you all? I know people have commitments, but look at me - I hardly have time to go to the toilet let alone take half a day off. I have managed to organise my time so that I can take half of Sunday off. The next thing is that some people have pointed out that the main meeting is Very Boring, so at this meeting I'm going to experiment with some demos and liven it up a little. The next point is the SIGs (Special interest groups) are there for you, people give up a lot of their spare time to organise these separate and detailed meetings and they are very informative. If you don't come to the meeting every month you don't know how helpful they are. My games SIG for example is very informative - I often say "so that's how you get past that monster". Sometimes I go to some of the programmers SIGs and say "Ah that's how you get sprites to move". I know most people join the club just for Workbench, but they will never get the one-to-one basis you get at the meeting and the SIGs. You can ask others who may have experienced your problem before how they dealt with it or how they remedied the situation. You should at least come and see what does go on at the meeting instead of saying "I pay my $20 a year" and that's that. We have much more to offer on a one-on-one basis than a smattering of information in the newsletter.

The last thing is that I would like those who come to the meeting regularly, or others for that matter, to bring along another interested party who is not a member of the club. He/she doesn't have to own an Amiga or want to own one, just see if he/she would like to come along and see what we're on about. You never know, they just might like us! If they are thinking about buying a computer, tell them to come along to one of the meetings and talk to us before they buy. Remember we are a non-profit organisation and we show no biases to any dealer, we just state who we feel provides the best after sales service and whose deals are the best. We can also indicate the best price you should pay etc... We have a wealth of knowledge when you put us all together, and if we don't know, then maybe we can find out for you.

So, I would like to see more PARTICIPATION; for example please show some respect to our editor and write more articles. He already gives up 2 of his weekends a month to put this fantastic screed together and the other weekend is spent at the meeting. So please give him something to print!

The bottom line is it's YOUR club make of it what you want, but if you want to make this club great, as I would like to see it, we need more participation and we could do with a few more members.

See YOU at the meeting!

Public Domain Software Order Form

Mail to: Amiga Users Group, PO Box 48, Boronia, 3155, Victoria

Disk Numbers:							

Don't forget to specify collection name, ie Fish, Amigan, Amicus, etc

Disks supplied by Amiga Users Group @ $8 each	$
Disks supplied by member @ $2 each	$
Club Use Only:	Total $
Member's Name:	Membership #:
Address:	
	Postcode:

Newsletter Back Issue Order Form

Mail to: Amiga Users Group, PO Box 48, Boronia, 3155, Victoria

Issue Numbers:							

Be patient, we may have to reprint some issues to fill your request

Number of issues ordered @ $2 each	$
Club Use Only:	Total $
Member's Name:	Membership #:
Address:	
	Postcode:

Application for Membership of The Amiga Users Group Inc

Membership is $20 per year. Send your cheque to: Amiga Users Group Inc, PO Box 48, Boronia, 3155

Surname: _____   Details on this side are optional

First Name: _____   Year of birth: _____ Which model Amiga: _____

Address: _____   Occupation: _____

_____ Postcode: _____   Interests: _____

Phone Number: _____ STD Code: _____

Where did you hear about AUG: _____

_____   Dealer's Name: _____

_____   Dealer's Address: _____

Signed: _____ Date: _____

If admitted as a member, I agree to abide by the rules of the Association for the time being in force.

Club Use Only	Date	Paid	Rcpt #	Memb #	Card Sent

AUG meets on the
third Sunday of each month

Monash University is in Wellington Road, Clayton. See Melways Map 70, reference
F10. Melways map 84A shows the University Campus in details. I've drawn a huge
arrow on the map below to show where the Rotunda is. The best place to park
your car is the car park area between Wellington Road and the Rotunda. The
entrance to the Rotunda is virtually at the point of the arrow.
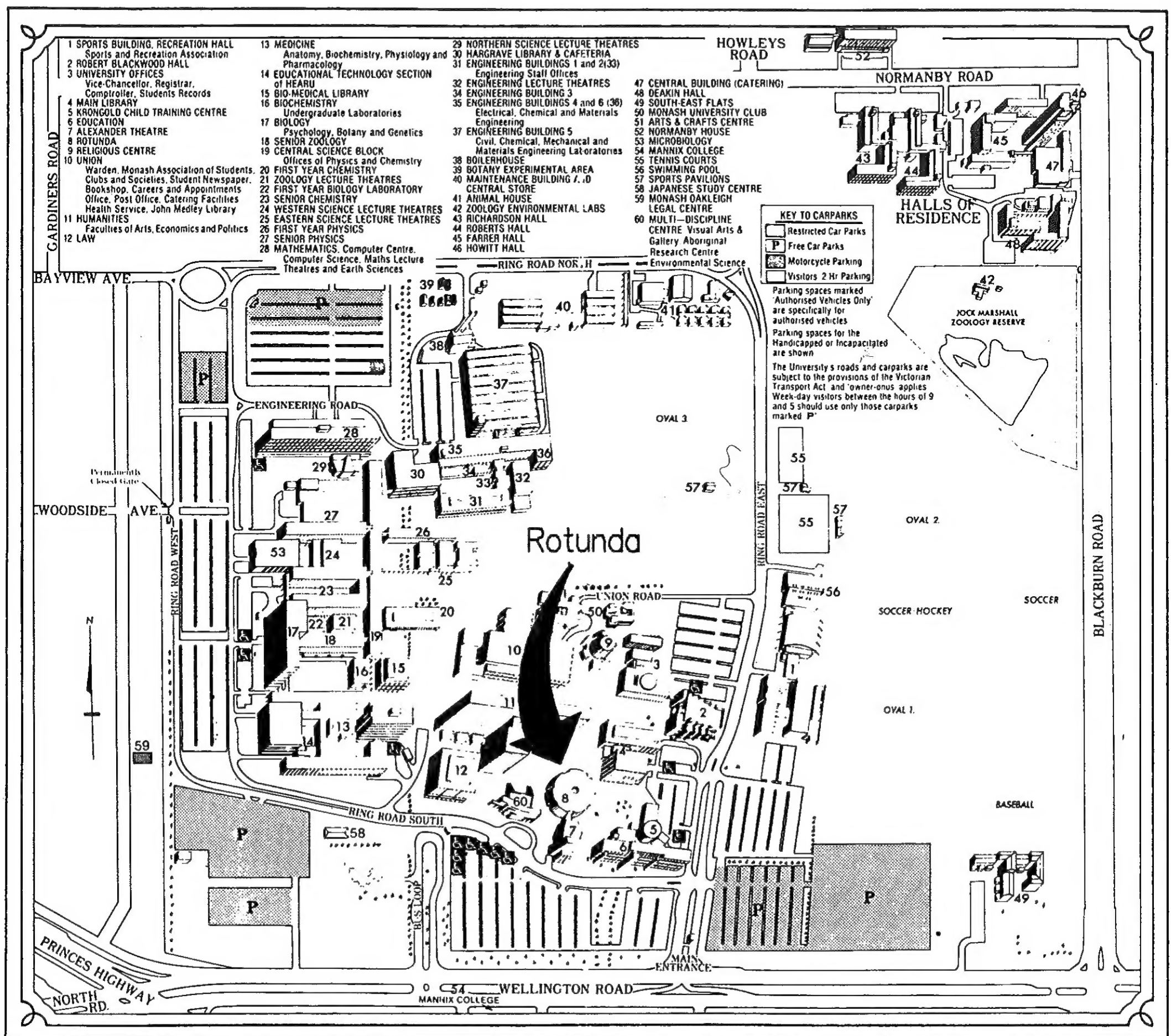
BY PUBLIC TRANSPORT . . . The simplest method is to take a train from Flinders Street or Loop stations on the Dandenong/Pakenham line to either Huntingdale or Clayton. Buses run from these stations to the campus or there is a taxi rank at Clayton. With suitable connections the trip takes about 45 minutes — but it can take longer! An inner neighborhood ticket will take you all the way via Huntingdale station and the bus, but you will need to purchase a comprehensive ticket for the trip via Clayton, which encompasses two neighborhoods. The campus is also served by buses from Box Hill, Blackburn, Belgrave, Chadstone, Jells Park-Glen Waverley, Dandenong-Mulgrave, Oakleigh and Elwood.

FROM THE CITY BY CAR . . . An easy route is along St Kilda Road or Kingsway/ Queens Road and then on to Dandenong Road. The campus's tall Menzies Building comes into view a kilometre or so before the left turn into Wellington Road on which the main entrance is located. Allow 40-50 minutes for the trip. Drivers should note that restrictions apply in some car parks weekdays 9 a.m. to 5 p.m. and fines **do** apply. There is ample unrestricted parking and, closer to buildings, designated two hour visitor car parks — check the map or ask at the Gatehouse.